STOCKTOPIA - Real Time Stock Index Dashboard and Anomaly Detection System

Authors: More Ajinkya, Walunj Alisha, Atre Atharva *Purdue University Fort Wayne*

Supervisor: Prof. Beomjin Kim *Purdue University Fort Wayne*

ABSTRACT: *Stocktopia is a real-time stock index dashboard designed to enhance market surveillance and investor awareness by providing immediate access to stock index fluctuations and anomaly detection. The platform utilizes Python, React, and Chart.js to deliver an intuitive user interface that shows live data for major indices like the Dow Jones, NASDAQ, and SP 500.*

The system not only displays real-time data but also alerts users to significant price changes based on predefined thresholds. These thresholds are informed by historical data analysis and trend identification using an LSTM neural network model. The backend, developed with Flask and PostgreSQL, manages data acquisition and processing, ensuring a seamless and responsive experience.

Stocktopia's continuous monitoring and alert system promote market transparency and enable investors to make well-informed decisions quickly. This abstract highlights the core functionalities, technologies, and the significant impact of Stocktopia on enhancing financial decision-making in dynamic market conditions.

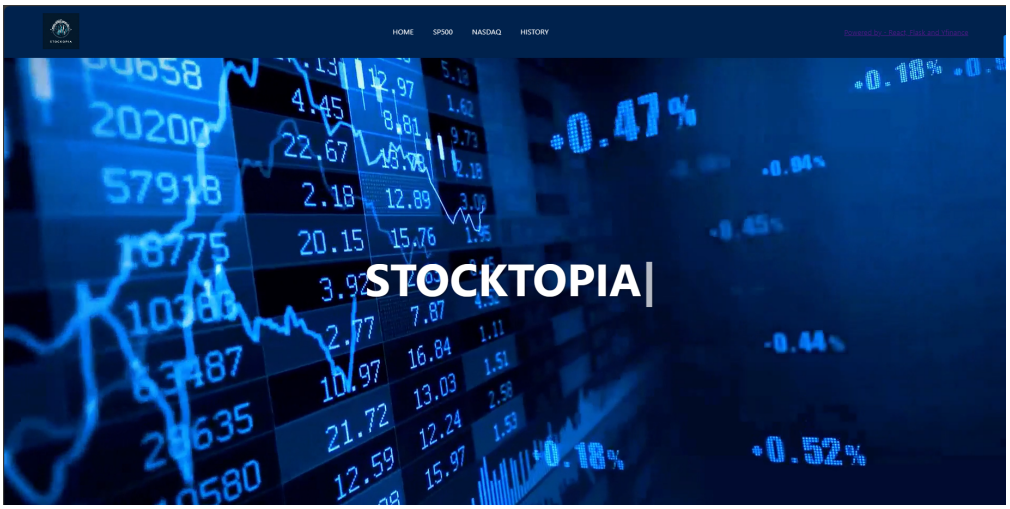


Figure 1: Stocktopia - An Real-time stock market analysis tool

1 INTRODUCTION

Stockpia is a real-time stock index dashboard designed to provide users an accessible way to monitor market trends and detect anomalies. It features visualizations of stock index data and alerts users when stock prices significantly deviate from a predefined range. This indicates potential market fluctuations. The indices included for exploration and data analysis are the Dow Jones, Nasdaq, and SP 500.

1.1 Purpose

Stocktopia serves as a comprehensive solution catering to the needs of investors across all proficiency levels, from beginners taking their first steps into the market to experts navigating its complexities. Its primary objective lies in furnishing users with an intuitive platform for closely monitoring market trends in real time. By offering a dynamic interface that provides up-to-the-minute updates on key market indices such as the Dow Jones, Nasdaq, and SP 500, Stocktopia empowers investors to make informed decisions about their investments. We have listed few key points

- Design an interactive dashboard that is easy to use;
- Implement logic to handle live real-time data;
- Provide notification with the parametric reason to trigger these notifications;

1.2 Background

Through our research of available solutions in the market, we've identified several options:

- **Investor Dashboard:** These platforms offer an intuitive dashboard interface for presenting stock market data. However, they encounter challenges in providing real-time calculations.
- **Market Watch:** Similar to Investor Dashboard platforms, Market Watch tools also present market information. However, they lack the functionality for users to define specific parameters for receiving notifications based on changes.
- **Trading View:** This website offers various visualizations of market data. However, users are unable to customize parameters to establish notification triggers.
- **Standalone Applications:** Some standalone applications tailored for stock market traders are available. However, they may lack comprehensive functionality, necessitating continuous monitoring by users to detect price fluctuations.

2 Proposed Solution

2.1 Solution

Our proposed solution involves analyzing historical data through Python and advanced data analytics techniques. Leveraging modern development tools such as React, PostgreSQL, Flask, and Node we propose the creation of a dashboard. This dashboard will allow users to input threshold values and will utilize Python to monitor if these values are exceeded. It will also generate alerts if there are price changes observed twice within the last 10 minutes, including the percentage change in price.

2.2 Functional Requirement

- **Real-Time Data Fetching:** The system should be able to grab the latest stock index information directly from specific financial markets as it happens.
- **Alert System:** Stocktopia needs to have a built-in feature that sends alerts to users when it spots any unusual activity in the stock market. These alerts could signal potentially important events happening in the market.
- **Historical Data Analysis**: Stocktopia should analyze historical data from as far back as 1995 up to the present year for three major indices: the Dow Jones, Nasdaq, and SP 500. This analysis helps users understand long-term trends and patterns in the market.

3 Data Description and Exploratory Data Analysis

3.1 Data Description

We've developed a Python Flask backend that retrieves data from the Yahoo Finance API. This backend is capable of fetching various parameters from the Yahoo Finance API. You can achieve the same functionality by running the following code snippet:

Listing 1: Code for loading live dataset

```
def get_data(symbol):
    stock = yf.Ticker(symbol)
    end_time = datetime.now() - timedelta(minutes=5)
    start_time = datetime.now().replace(hour=0, minute=0, second=0, microsecond=0)
    data = stock.history(start=start_time, end=end_time, interval="2m")
    return data


def get_live_data():
    data = get_data("^GSPC")
    X = [timestamp.strftime('%Y-%m-%d■%H:%M:%S') for timestamp in data.index]
    Y = data['Close'].tolist()
```

```
V = data['Volume'].tolist()
data_list = list(zip(X, Y, V))
return jsonify(data_list)
```

This Python function fetches live stock market data for the SP 500 index from the Yahoo Finance API. It extracts timestamps, closing prices, and trading volumes from the retrieved data. Then, it combines this data into a list of tuples representing each data point. Finally, it returns this data in JSON format, ready to be consumed by client applications.

Out[6]:

Date	Open	High	Low	Close	Adj Close	Volume
2024-03-04	2980.949951	3024.899902	2974.449951	3014.800049	3014.800049	5012210
2024-03-05	3011.550049	3014.800049	2972.100098	3000.399902	3000.399902	3553834
2024-03-06	2986.899902	3018.000000	2957.000000	3006.000000	3006.000000	3902838
2024-03-07	3005.949951	3006.199951	2951.100098	2957.850098	2957.850098	4157863
2024-03-11	2978.000000	2978.000000	2927.000000	2933.199951	2933.199951	5638565

Figure 2: Retrieved Data from Yfianace

3.2 Data Plots Used

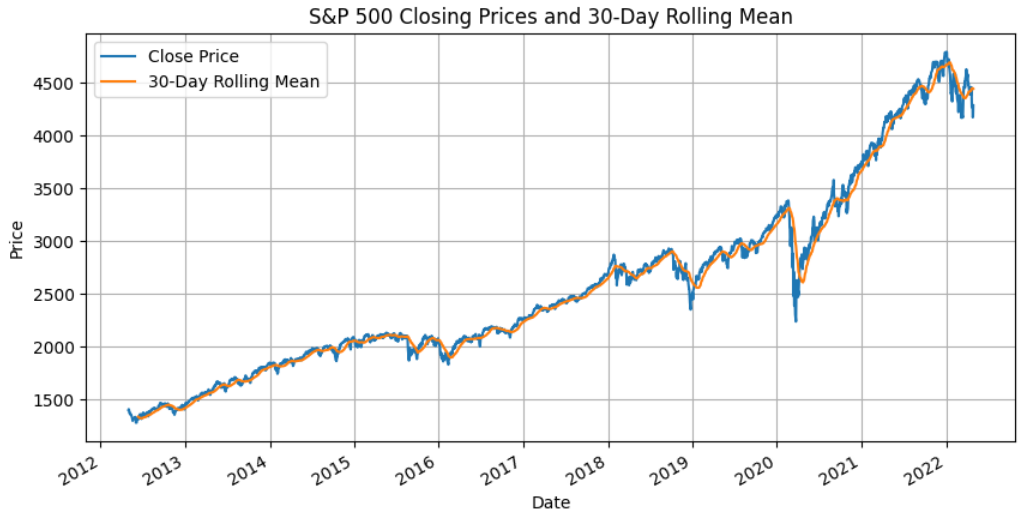


Figure 3: Line Chart Plot used for Visualization of SP500 Index

1. **Line Chart for Closing Prices and Rolling Mean**:
   - **Purpose**: This plot displays the daily closing prices of the indices (Dow Jones, Nasdaq, and S&P 500) along with their respective rolling mean values over a specified window size.
   - **Reason for Use**:
     - **Trend Visualization**: Line charts are suitable for visualizing trends over time. By plotting the closing prices and rolling mean values on the same chart, viewers can easily observe the trend of the indices' prices.
     - **Comparison**: Line charts allow for easy comparison between two or more series of data. Comparing the actual closing prices with their corresponding rolling mean values helps identify deviations from the overall trend.
2. **Candlestick Chart**:
   - **Purpose**: This plot visualizes the price movement of the indices (Dow Jones, Nasdaq, and S&P 500) using candlestick patterns.

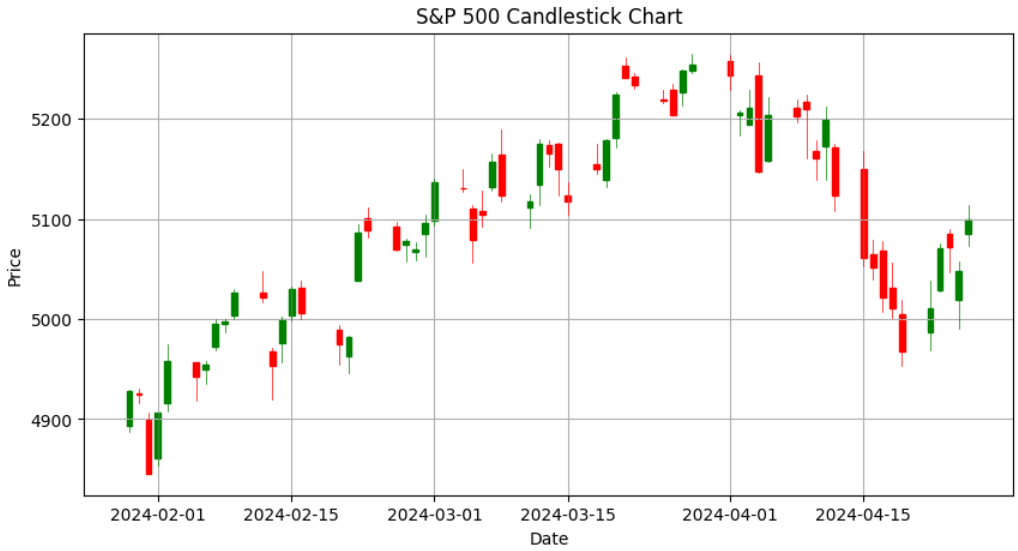


Figure 4: Line CandleStick Plot used for Visualization of SP500 Index

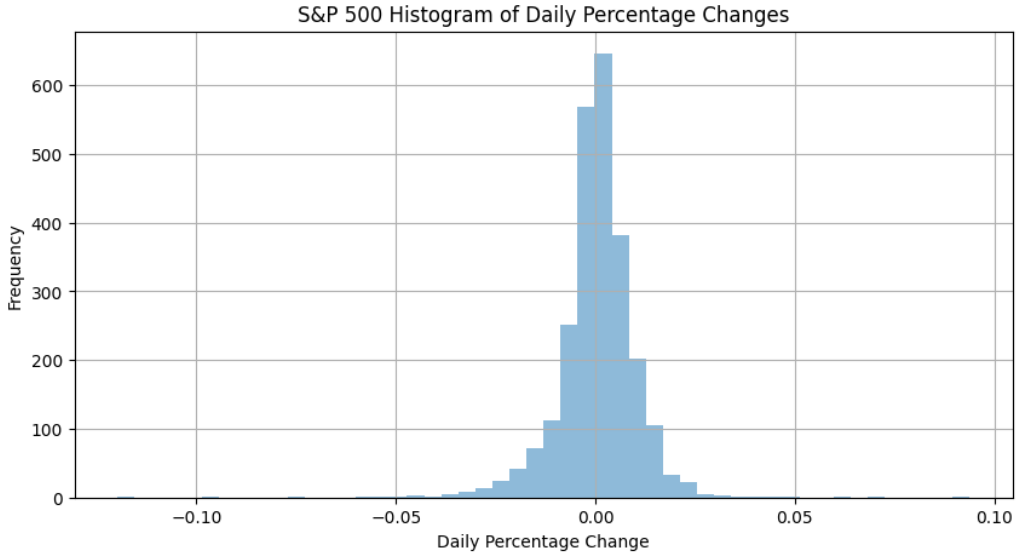


Figure 5: Line Histogram Plot used for Visualization of SP500 Index

- **Reason for Use**:
  - **Detailed Price Information**: Candlestick charts provide detailed information about the open, high, low, and close prices for each time period. This level of detail helps traders and analysts identify patterns.
  - **Volatility Analysis**: Candlestick patterns reflect price volatility and sentiment in the market.

3. **Histogram of Daily Percentage Changes**:

- **Purpose**: This plot showcase the distribution of daily percentage changes in the indices' closing prices.
- **Reason for Use**:
  - **Distribution Analysis**: Histograms are effective for visualizing the distribution of numerical data. The histogram helps assess the frequency and magnitude of daily price changes.
  - **Identifying Patterns**: By examining the shape of the histogram, traders and analysts can identify patterns in price movements, such as skewness which may inform and help in decision-based trading strategies.

3.3 Data Aggregation:

Historical daily closing prices of the SP 500 index were collected. Returns for each day were calculated based on the previous day's closing price. The data was grouped by calendar dates(irrespective of year), aggregating returns over the

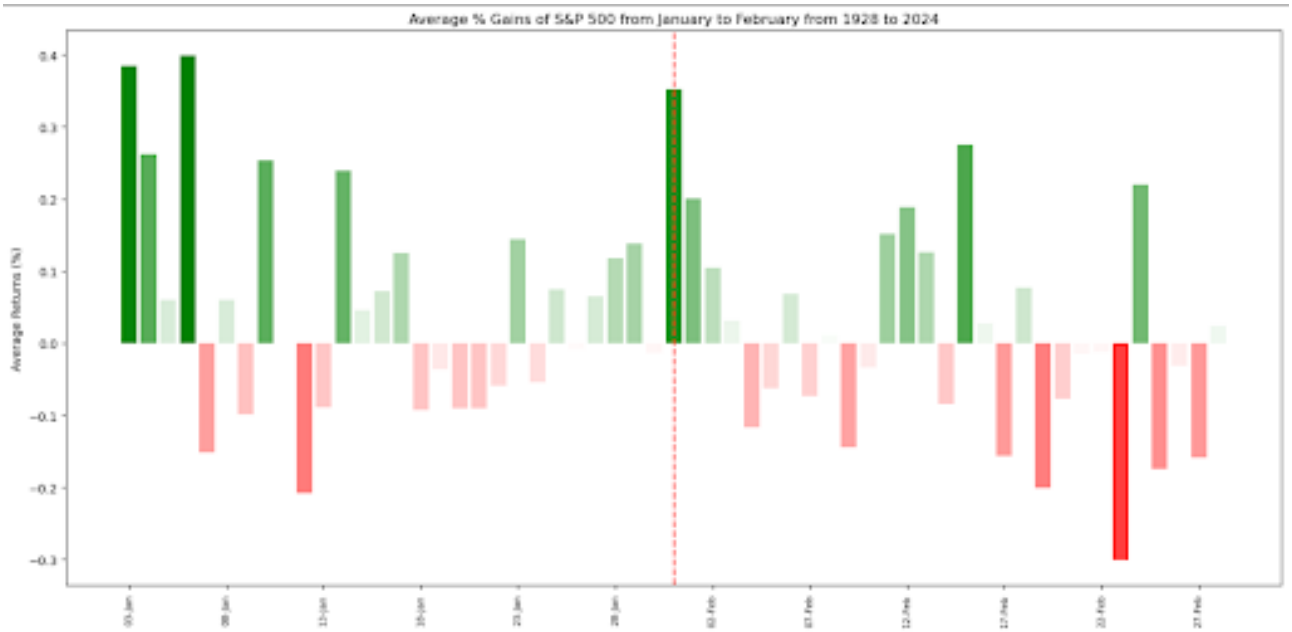
entire period of 1928-2023 Averages of these returns were computed to represent each specific day

3.4 Focus on Daily Returns:

The analysis emphasizes the average return for each day rather than cumulative performance. Daily average returns are highlighted, providing a granular view of market behavior. Usefulness for Analysis: The visualization enables analysts to discern patterns related to these specific months over the span of 1928-2023 It assists in strategic planning by indicating historical trends and potential future performance.

3.5 Monthly Gain Analysis

Jan - Feb Performance



As shown here, the Jan-Feb transition for the SP500 index has historically been very positive. There is also very strong return potential in the beginning of a year (Jan start), may be attributed to last quarter earnings release for companies (decembers), or the market being positive after Christmas - New years vacation.

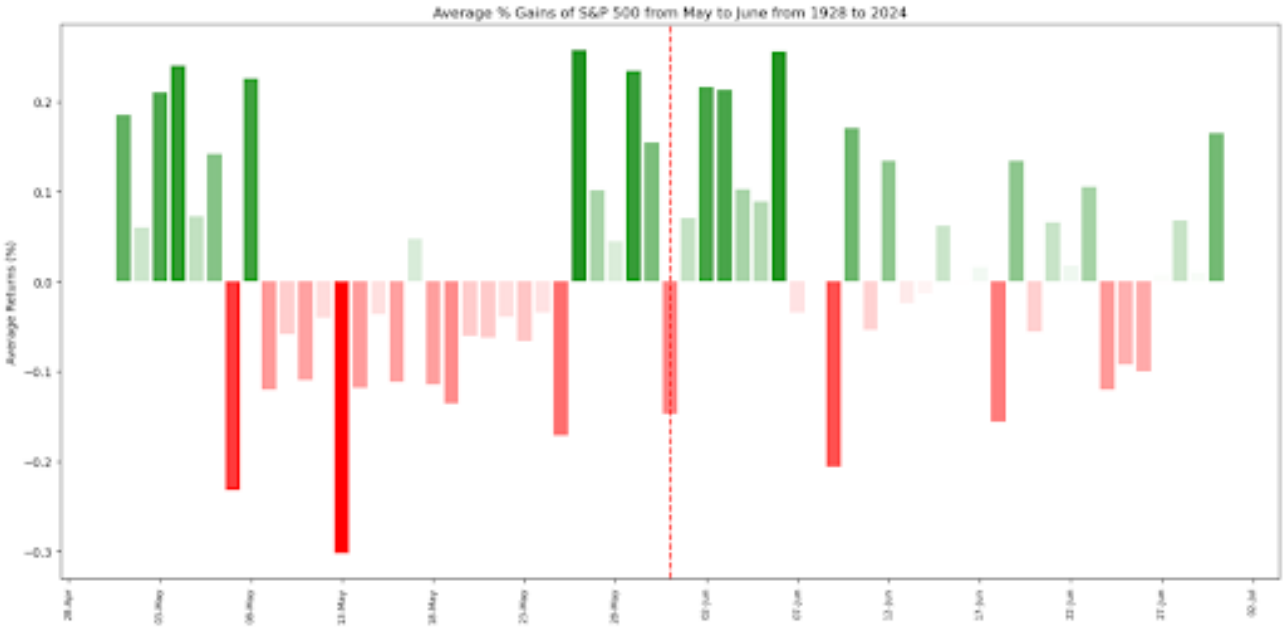
Feb - March Performance:
We again see the same trend of the index showing good return potential for Feb End - March beginning historically. Though we note that the market is volatile (especially in March end), could be attributed to expectations of Q1 earning release.

March - April Performance

This gives a very unique birds-eye view on March End to April Beginning data. This records a strong negative gain for the index historically. This may be attributed to the first earning-release of the year for public companies.

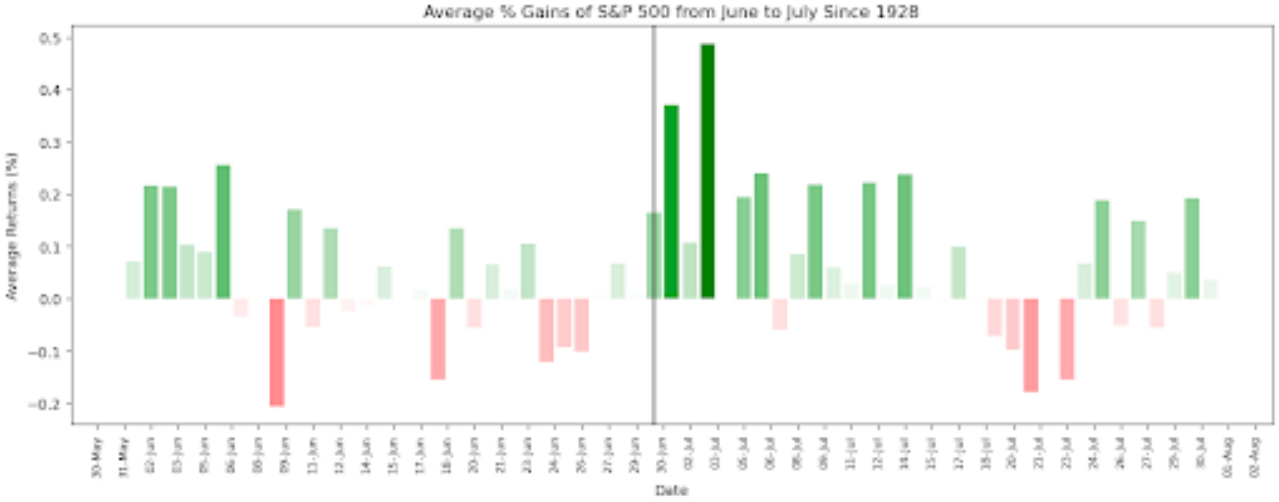
April End - to May Beginning again shows an overall positive return. April start especially, shows a good long stretch of positive returns. Did the companies reason with the stakeholders explaining their Q1 performance?

May - June Performance



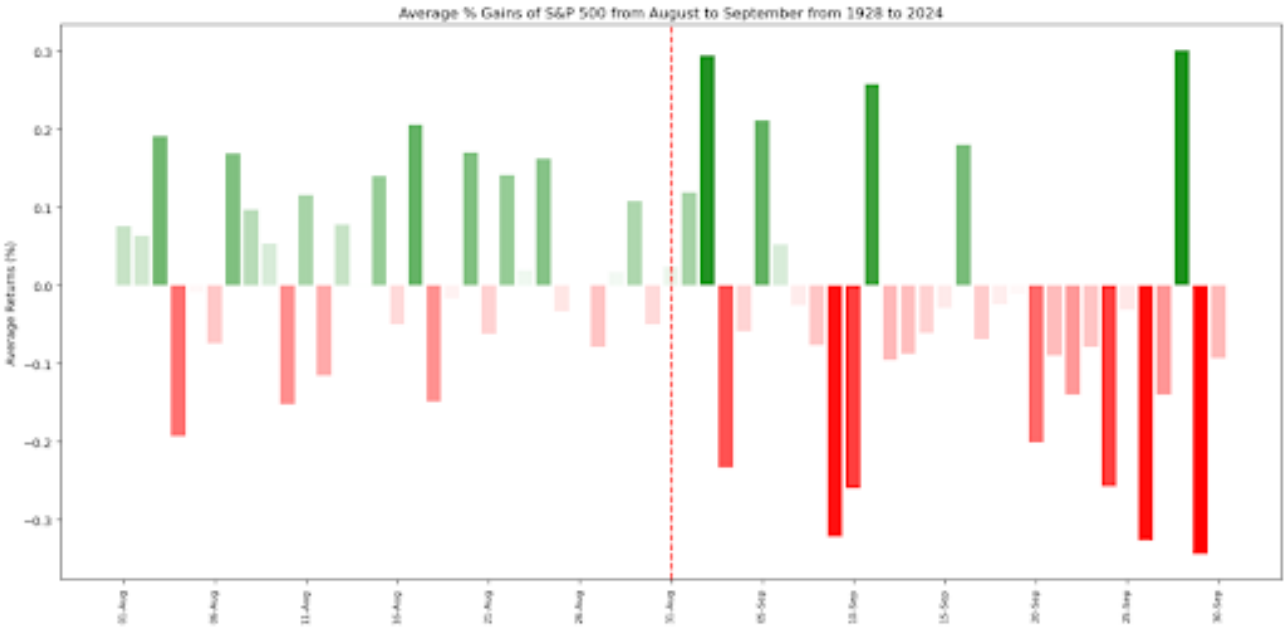
May End- to June beginning has an average negative return, but it is in a patch where the market is giving good positive returns historically. So , we would definitely recommend keeping an eye on the market for investing despite negative returns during this period. May 10-25th has a very strong negative returns patch.

June - July Performance



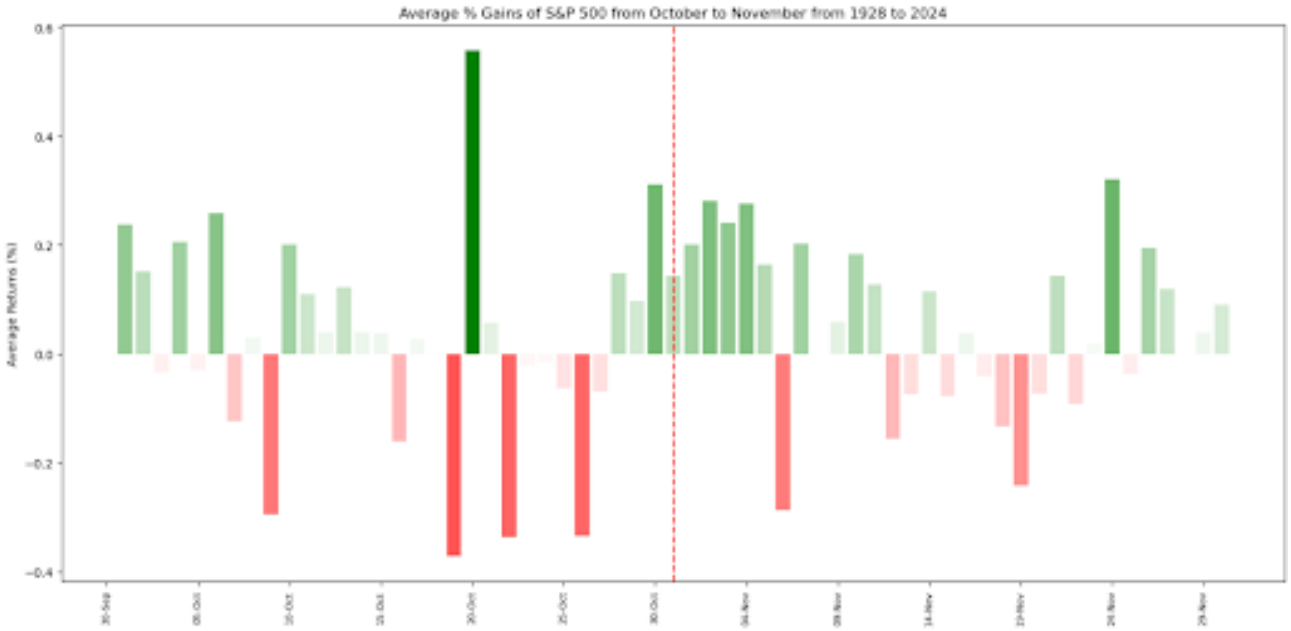
June End - July start also follows the trend of the good strong index performance when a new month starts. This patch has some of the least volatility in terms of possibility of good returns (No darker shade of red present). July end - August start also has good historical index performance, but very small. It means there are many years where the index was in negative gain, but the overall aggregate was slightly on the positive side. This overall period has historically been giving positive returns.

August - September Performance



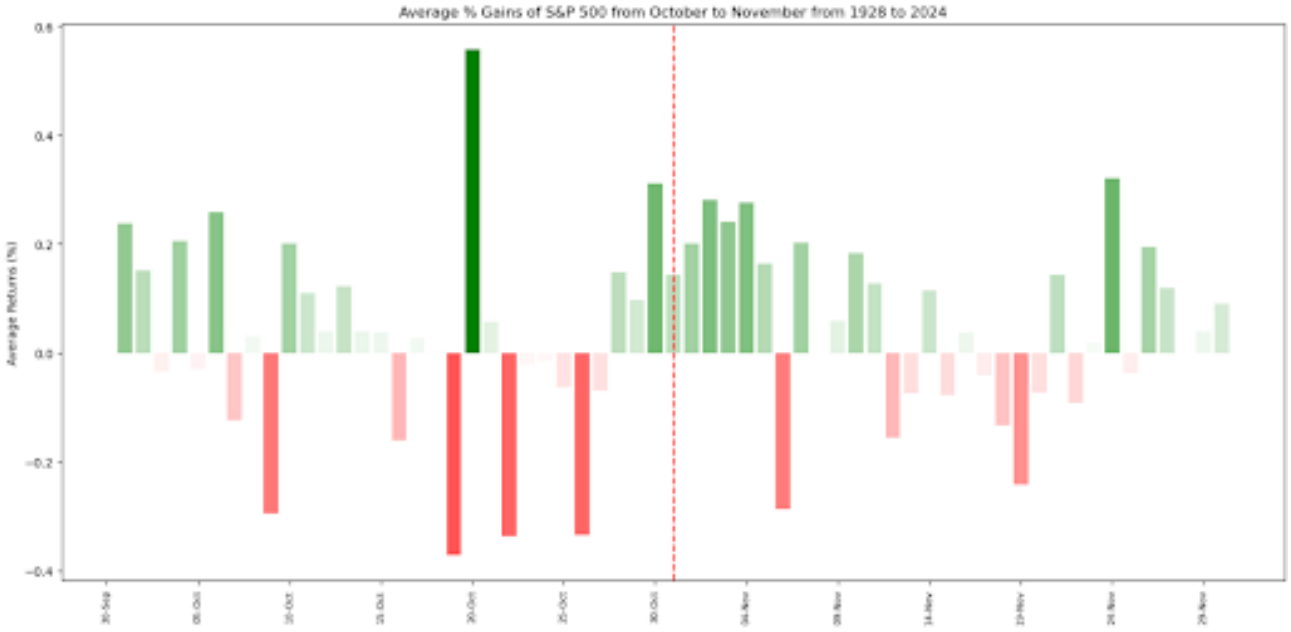
This period shows the most market volatility (dark greens and dark reds in quick successions). We believe that this can be attributed to the Q3 earnings call release. Some years may have bad earnings, while others may have been good.

September - October Performance



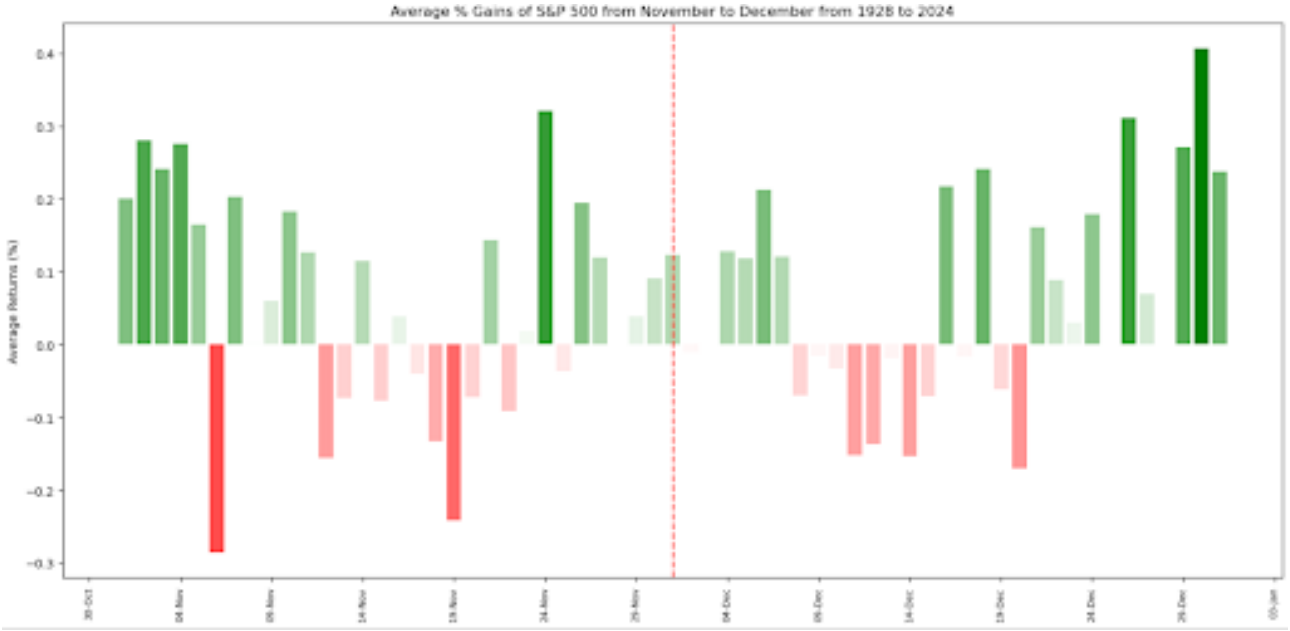
September to October period is a well-rounded fluctuation in gains/losses historically for S&P 500.

October - November Performance



October to November transition has historically been on a strong positive gain.

November - December Performance



November to December transition again is on a positive gain. Very strong December end performance continues well into January start (shown in first plot). This coincides well with the Q4 earnings call release for public compan

3.6 Insights from EDA

Seasonal Trends

The analysis highlights certain seasonal trends in the stock market, such as the positive performance during the transition from December to January, possibly influenced by year-end holiday sentiment and earnings releases.

Earnings Season Impact

The performance patterns around the beginning of each quarter suggest that earnings releases play a significant role in market behavior. Positive or negative surprises in company earnings could drive market movements during these periods.

Market Volatility

Certain transitions, such as August to September, exhibit higher volatility, possibly due to heightened uncertainty around quarterly earnings releases or other economic factors. Recognizing these periods of increased volatility can help investors manage risk more effectively.

Impact of External Events

External events such as regulatory changes, geopolitical tensions, or natural disasters usually influence market performance differently during various periods, in various years. Understanding how these events interact with historical performance could be a key extension of our work.

4 Anomaly Detection Model

April 2000: Dot-com Bubble Crash[1]

The dot-com bubble burst in April 2000, marking a dramatic end to the rampant speculation in technology-related stocks. During the late 1990s, investor enthusiasm for internet-based companies led to a stock market bubble. Many of these companies, known as ”dot-coms,” commanded huge market valuations despite often lacking sustainable business models or even significant revenue. When investor confidence waned, the NASDAQ composite, heavily laden with tech stocks, collapsed, dropping from its peak by over 75 percent by October 2002, erasing billions in market value.

September / October 2008: Wall Street - Lehman Brothers Crash[2]

In September 2008, Lehman Brothers, a venerable investment bank, filed for bankruptcy following a massive exodus of most of its clients, drastic losses in its stock, and devaluation of its assets by credit rating agencies. This event was a pivotal moment in the Global Financial Crisis and exemplified the troubles in the banking industry, which was overly exposed to poorly documented mortgages with declining value. The collapse triggered a global financial panic as it became clear that the financial system was more interconnected and vulnerable than previously understood.

March 2020: COVID-19 Crash[3]

The COVID-19 crash in March 2020 occurred as global markets reacted to the rapid spread of the coronavirus, which led to unprecedented public health responses including lockdowns and travel bans. Fears about the virus’s impact on economic activity and corporate profits led to a sharp decline in stock markets around the world. The SP 500, for example, saw its fastest drop into bear market territory in history, falling over 30 percent from its peak in just over a month. The crash was marked by extreme volatility and uncertainty about the pandemic’s long-term economic effects.

The Dow Jones Industrial Average experienced significant drops, reflecting the global economic uncertainty caused by the virus outbreak. On March 16, 2020, the Dow fell by 2,997 points, marking a 12.93 percent decline, which was the largest single-day point drop in its history up to that point. Other significant declines occurred on March 12, with a drop of 2,352 points (a 9.99 percent decrease) and on March 9, where it fell by 2,013 points, a 7.79 percent decline (Nasdaq) (Wikipedia).

This period was marked by extreme volatility, with the Dow Jones plunging 38.3 percent from its February high to its March low, showcasing one of its steepest monthly declines on record. These fluctuations indicated the market’s rapid reaction to the unfolding pandemic and the global economic slowdown.

5 The Dashboard

5.1 Backend

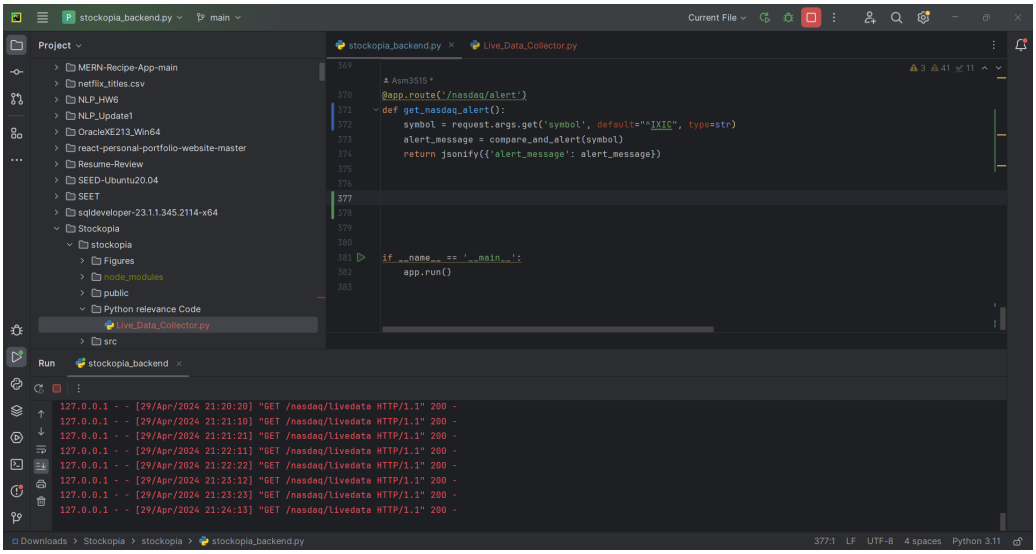


Figure 6: Backend Running on pycharm IDE and we can see the request and response being received and sent continuously

The backend system developed using Flask and Yahoo Finance (yfinance) libraries is designed to provide real-time and historical stock market data. This system is capable of delivering data through various API endpoints that handle live, recent, and metric-driven data requests for two stock indices i.e Nasdaq and SP500. The application has Cross-Origin Resource Sharing (CORS) to ensure it can be accessed from react and node for frotend.

Comparison and Alert Functionality

The backend code includes an alert system that compares average prices over different time frames (7 days vs. 1 month) to detect significant market changes. If the market has undergone a rise or fall, as specified by the user's threshold, the system will generate an alert message. This functionality is crucial for monitoring market volatility and advising users of significant changes in stock indices like the S&P 500 or NASDAQ.

Live Data (`/livedata`)

This API returns real-time trading data for the S&P 500 index or NASDAQ. It formats the data into a list of timestamps, closing prices, and trading volumes. This endpoint is designed to support real-time chart displays, making it extremely useful for applications that require up-to-the-minute data visualization.

Seven Days Data (`/7days`)

This API provides a summary of the past week's trading data for the S&P 500. It includes metrics such as average price, volume changes, and market trends. This endpoint is particularly valuable for weekly market analysis and review, helping users to track market performance over a manageable time frame.

Market Check (`/checkmarket/<percentage_change>`)

This dynamic endpoint checks if the market has risen or fallen by a specified percentage. It is highly customizable, allowing users to set their own thresholds for alerts. This feature is designed to trigger notifications based on user-defined criteria, enhancing the application to meet specific user needs which solution to date wont provide.

5.2 Frontend

The frontend component of the NASDAQ Live Dashboard is built using React, a popular JavaScript library for building user interfaces. The system utilizes Chart.js for visualizing real-time data through graphical representations, specifically line charts, to display NASDAQ and SP 500 indices data. The application is styled using a dedicated CSS file and integrates widgets for additional data insights.

Figure 7: Backend Running on pycharm IDE and we can see the request and response being received and sent continuously

Functionalities

Live Data Visualization

The frontend application utilizes React's `useEffect` hook to perform several critical tasks:

- It automatically fetches live data from the backend every minute from the endpoint `http://127.0.0.1:5000/nasdaq/livedata`.
- Updates the `latestData` state with the fetched data.
- Sets the `isDataFetched` flag to true upon successful data retrieval.

Dynamic Chart Configuration

The chart is dynamically configured to enhance clarity and user experience:

- Configured with responsive settings to adapt to different screen sizes.
- Styled to provide clear visualization, including bold labels and colors background for better readability.
- Utilizes `moment.js` for time formatting to format the x-axis labels in a user-friendly "minute" format.

Alert Functionality

The dashboard includes an interactive alert functionality:

- Users can set a custom percentage change threshold through an input field. This setting triggers an alert if the market fluctuates beyond the specified threshold.
- The `submitPercentageChange` function sends a request to the backend endpoint `http://127.0.0.1:5000/nasdaq/checkmarket/\{percentageChange\}`. Based on the response, an alert is shown to the user, notifying them of significant market movements.

Automatic Alert Checks

To enhance interactivity and utility, the application sets an interval to automatically check for market alerts:

- The check is performed every 2 minutes when a percentage change threshold is set.
- This feature ensures that the user is continuously informed about significant market changes without needing to manually refresh or query the data.

6 Live Demonstration Videos

For a live demonstration of the dashboards, please visit the following link:

`https://github.com/Asm3515/stock-pia/blob/main/FInal_Se_Threshold_Part2.mp4`

7 Team Member Contribution

This section outlines the specific contributions made by each team member to the Stocktopia project.

Team Members	Contributions
Alisha Walunj	Worked on the UI integration, focusing on ensuring that the frontend effectively communicates with the backend services and provides a seamless user experience.
Atharva Atre	Worked on historical data analysis and the anomaly detection model, developing algorithms to analyze past market data and detect significant deviations.
Ajinkya More	Worked on real-time data acquisition, storage, backend, and plotting, ensuring timely fetching, secure storage, and effective visualization of stock market data.

Table 1: Contributions of each team member to the Stocktopia project.

8 Conclusion

In today's financial markets, significant and abnormal changes often occur rapidly, where reliance on static data analysis and manual tracking can be time-consuming and inefficient. Such traditional methods frequently result in missed opportunities or incorrect interpretations of market trends. Given these challenges, there is a clear and pressing need for a dynamic and automated solution to monitor and analyze market behavior effectively.

The project Stocktopia helps people not only to view the graph but also triggers an alert for the users when the index value crosses the threshold value. This innovative functionality enhances user engagement by providing timely and critical information that could impact investment decisions.

9 Lessons Learned/Challenges Experienced

9.1 Lessons Learned

- Handling real-time data is quite challenging due to the need for rapid processing and immediate response.
- For historical data, it is crucial to maintain high standards of data integrity and accuracy. Proper data management practices are essential to ensure the reliability of predictive analytics used in the project.

9.2 Technical Challenges

- **Historical Data Accuracy:** Gathering and utilizing extensive historical data to understand market behavior for threshold calculation was a significant challenge. We employed an LSTM model for predictions, necessitating thorough cleaning and preprocessing of data to ensure model accuracy and reliability.
- **Uncertain Market Behavior:** The unpredictable nature of the stock market, characterized by frequent minor fluctuations and occasional significant shifts, posed a challenge in setting effective threshold values. These thresholds had to be adaptive and responsive to real-time market conditions to trigger alerts appropriately.
- **Threshold Calculation:** Determining the correct thresholds based on historical data while aligning them with current market trends was complex. It required a delicate balance to ensure the model's sensitivity was optimal for real-world application.
- **System Integration:** Coordinating the integration of various components handled by different team members, such as real-time data handling, historical data analysis, and the frontend presentation, was challenging. Ensuring seamless data flow and functionality between these components was critical for the project's success.

- **API Rate Limiting:** Dealing with API rate limits was a critical issue, especially in a real-time data environment. Limited requests could hinder the ability to refresh data promptly. We chose Yfinance for historical data fetching, which offered a relatively generous limit of 2000 requests per hour, helping to mitigate this issue.

9.3 Limitations and Recommendations for further work

We have demonstrated a real-time system that can perform complex calculations on data and can plot real-time data (Visualise real-time Data) which is hard to do with a limited number of resources and constrained time available but this system has a massive delay in the real-time of about 5 Minutes form the original time which is the byproduct of limited computational power.